NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

DESIGNING REALISTIC HUMAN BEHAVIOR INTO MULTI-AGENT SYSTEMS

by

Chad F. Hennings

September 2001

Thesis Advisor:	John Hiles
Thesis Co-Advisor:	Rudolph Darken

Approved for public release; distribution is unlimited

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*

2. REPORT DATE
September 2001

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE: Title (Mix case letters)
Designing Realistic Human Behavior Into Multi-Agent Systems

5. FUNDING NUMBERS

6. AUTHOR(S) Hennings, Chad F.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(maximum 200 words)*

As Multi-agent systems advance toward moving virtual humans such as modeled infantry soldiers around a virtual environment for modeling and simulation purposes, an important factor to be considered is how the agent internalizes and reacts to its environment. One method to simulate this sensory perception and the construction of generalized internal knowledge is the symbolic reactive agent architecture. This architecture utilizes symbolic constructive agents to internalize and symbolically represent the outside environment within the agent and reactive agents to decide what course of action will be taken next based on this internal environment. This type of architecture also lends itself well to putting variability and non-homogeneity into different agents by controlling the level of hindrance or interference that the agent utilizes when constructing this inner environment. A simple path-finding task was used to determine the overall utility of this architecture with respect to truly representing human performance in cognitive tasks. Humans as well as different simulated agents were put through the same task in their respective environment and their results were compared. A concept called the bracketing heuristic was also utilized to determine whether the model may translate well to general path-finding tasks.

14. SUBJECT TERMS Agent Based Modeling, Multi-agent Systems, Bracketing Heuristic, Symbolic Reactive Agent Architecture.

15. NUMBER OF PAGES 72

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
Unclassified	Unclassified	Unclassified	UL

THIS PAGE INTENTIONALLY LEFT BLANK

DESIGNING REALISTIC HUMAN BEHAVIOR INTO MULTI-AGENT SIMULATION

Chad F. Hennings
Lieutenant, United States Navy
B.S., Illinois Institute of Technology, 1994

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN MODELING, VIRTUAL ENVIRONMENTS, AND SIMULATION

from the

NAVAL POSTGRADUATE SCHOOL
September 2001

Author: ____________________________________
Chad F. Hennings

Approved by: ____________________________________
John Hiles, Thesis Advisor

____
Rudolph Darken, Co-Advisor

____
Rudolph Darken, Academic Associate
Modeling, Virtual Environments, and Simulation Academic Group

____
Michael Zyda, Chairman
Modeling, Virtual Environments, and Simulation Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

As Multi-agent systems advance toward moving virtual humans such as modeled infantry soldiers around a virtual environment for modeling and simulation purposes, an important factor to be considered is how the agent internalizes and reacts to its environment. One method to simulate this sensory perception and the construction of generalized internal knowledge is the symbolic reactive agent architecture. This architecture utilizes symbolic constructive agents to internalize and symbolically represent the outside environment within the agent and reactive agents to decide what course of action will be taken next based on this internal environment. This type of architecture also lends itself well to putting variability and non-homogeneity into different agents by controlling the level of hindrance or interference that the agent utilizes when constructing this inner environment. A simple path finding task was used to determine the overall utility of this architecture with respect to truly representing human performance in cognitive tasks. Humans as well as different simulated agents were put through the same task in their respective environment and their results were compared. A concept called the bracketing heuristic was also utilized to determine whether the model may translate well to general path-finding tasks.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

The author would like to acknowledge Professors Rudy Darken and John Hiles for all their support and guidance. Their knowledge and excitement for the subject this thesis covers was truly inspiring. Lastly, my wife Caroline and daughter Dannica deserve my immense love and gratitude for all their love and support. I am truly blessed.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. REPRESENTATION OF HUMANS IN MULTI-AGENT SYSTEMS

Multi-agent systems have started to model actual human representations in a real world environment. Many simulations modeling infantry personnel especially have been successfully engineered. The simulations hold great promise in the field of high-resolution modeling. Not only can logic factors be taken into consideration when modeling the decision-making process of the agent, but also personality and emotional factors have been deemed as just as important ingredients to ensure that the agents exhibit "human" characteristics.

One component of agents that has lagged behind other considerations is how the agent perceives its environment. This perception consists of two things. The first is how accurate or inaccurate the incoming information should be modeled to represent such phenomena as the "fog of war", or how trustworthy and clear the information that is taken in by human beings actually is. The second is how do humans store information in their minds for later use. What are the ways in which humans use the data they have taken in to formulate an overall picture of what the world around them is so that they may take prudent and effective action?

Many types of multi-agent systems use a sensor radius to determine when an agent is given the information that would ordinarily be taken in through our human senses. Once an object or other entity has entered the agent's surrounding radius, the agent is given the informational parameters that it needs from this other object or entity in order to take decisive action on what to do next. Some simulations go as far as to use probabilistic outcomes as to decide when or how close an agent has to be to an object before it can actually see it and study its information. This representation can yield good results if statistical information exists for such exercises, but if this data does not exist then the function must be modeled accordingly.

While the above description provides an example of how the agent takes in information from its environment, it fails to describe how an agent internalizes this data for later use. The relationship that this data forms with other pieces of information that has been gleaned from the environment is not modeled.

B. USE OF HYBRID SYMBOLIC REACTIVE ARCHITECTURES

Recently, agent architectures have been proposed that utilize a symbolic part of the agent that samples and internalizes information from the agent's environment. This internalization and representation of the environment is called the agent's inner environment. This internal representation allows the agent to form relationships and organizations of the data that the symbolic portion of the agent internalizes.

In order for the symbolic portion of the agent to be able to construct an imperfect inner environment, some sort of interference mechanism must be applied to the data that comes into the agent. This interference can be a function that filters the information or some additive or multiplicative process that distorts the value or type of the raw data coming into the agent.

The other portion of the agent is its reactive portion. This is the decision making part of the agent. Decisions that are arrived at by this portion of the agent are formed using the inner environment that the agent constructs in its symbolic portion. Thus, the worse the agent's inner representation of the outside environment then the worse the decisions made by that agent will most likely be.

By using this hybrid architecture several ways to further 'humanize' or inject variability into the simulation are possible. For example, some agents, through such factors as natural ability or emotional stability, can have more trouble accurately depicting correct representations of outside generalized knowledge than others. Different tasks that the agent performs can also have more interference injected into the raw incoming data to represent tasks that are much harder to perceive than others.

C. HUMAN VERSUS AGENT COMPARISONS

In order to insure that the models that are used to create the interference in the agent's internal environment reflect real human performance when coupled with the reactive portion of the agent to produce behavior, it is useful to compare actual human performance with those of the agents. The goal here is not necessarily to create the same symbolic relationships that humans do when they perform cognitive tasks, but to create a mechanism that accurately reflects human performance.

In the general human population, there will be those people who have a better affinity for certain tasks. In the way that an agent's perceptual ability can vary, it is possible to lower or raise the level of interference the agent adds to the incoming data. This process is important because it adds the variability into simulation that exists in human subjects in the real world.

D. CREATING GENERALIZABLE MODELS

Once the average data for the agents reflects that of the actual human subjects, it is important to determine if the model for perceiving and acting on the environment is valid for generalized tasks of this nature or just for the task studied alone. There are different ways one can do this. One way is through a method used by David Kieras and David Meyer at the University of Michigan on cognitive models such as EPIC called the bracketing heuristic.

The bracketing heuristic basically states that models need to be able to reflect good or poor performance depending upon the type of person that is present. The first person is your conscientious user that puts his or her all into the task at hand. The second type of person is one that goes through the motions of the task or just has a low affinity for it. If actual human data falls somewhere in between these two brackets then you have a model that may not only do a good job of showing variability among different agents for the particular task, but may also perform general tasks well that are similar in nature to the exact one modeled.

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. EARLY SIMULATION IN OPERATIONS ANALYSIS

Since people have had to make important decisions that affect the well being of others as well as themselves, decision makers have utilized information to make informed and accurate decisions. Research and knowing the facts surrounding a particular decision were one of the first methods of ensuring accurate and sensible decisions were made. As decisions and decision-making became more complex, people yearned for a more complete way of gathering data together and judging what outcomes their decisions might lead to. Once research and organization of data achieved its pinnacle of usefulness, decision-makers desired a way to get a best guess or estimate of what outcomes their decisions may yield. The field of Operations Research was born. Hubert Dreyfus [Ref. 1:p. 8] describes how his brother, Stuart Dreyfus, and a famous mathematician, Richard Bellman worked to develop this young discipline, which enabled mathematical models to utilize the computer as a tool to determine the most optimal decision for various problems faced by generals, public policy-makers, and industrial planners.

1. Use of Deterministic Models in Decision Making

Personnel in the field of Operations Research started utilizing the simplest mathematical models known at the time. These models were known as Deterministic Models. Deterministic Models are defined as models whose output is "determined" once the set of input quantities and relationships in the model have been specified [Ref. 2:p. 6]. Even though it may take a computer many hours to compute the answers to a set of deterministic equations, the answer will be the same every time. Thus, there is no variability in the answer. In the realm of combat modeling, the simplest group of deterministic equations known is the Lanchester equations.

a. Lanchester Equations

In 1914, F.W. Lanchester introduced a set of coupled ordinary differential equations used to determine the outcome of a conflict based on the attrition factor, or how many entities are killed or destroyed in a particular time step and the number of enemy entities that exist who “kill” or destroy these entities. The equations were grouped into two categories, one for ancient warfare and one for modern warfare. The equations for modern warfare are listed below along with their respective meanings:

Equation 1:

$$dR/dt = -\alpha B\, B(t), \qquad R(0) = Ro$$

$$dB/dt = -\alpha R\, R(t), \qquad B(0) = Bo$$

where: dR/dt - attrition rate of R forces

dB/dt - attrition rate of B forces

αB - attrition coefficient for B forces

αR - attrition coefficient for R forces

The equations for ancient warfare are similar to the ones listed above. The only major difference is that the number of own forces is multiplied times the attrition coefficient in addition to the number of the enemies’ forces.

The attrition coefficients represented in the above equations by the symbol α represent the rate at which R forces are killed by B forces and vice versa. Thus, the attrition rate of the R forces and B forces are based on how many of the opponents’ forces are left and how good they are at killing the other’s forces.

Lanchester equations were a good starting point for combat modeling. They take the basic ideas such as how numerous an armies’ forces are and how effective they are at destroying the enemy. However, this method of combat modeling has come under fire for a variety of reasons. Critics say the equations represent combat in a very homogeneous nature. Attrition coefficients cannot vary due to considerations such as morale or supply superiority. There is no representation of force movement, terrain issues, or command and control issues.

b. Variations on Lanchester Equations

In order to alleviate the above criticisms of Lanchester equations many variations were made in order to simulate more realistic battlefield conditions. For example, Hembold Equations varied the attrition coefficient by taking the effectiveness of how an army concentrated its forces into consideration. This attempted to ease the criticism of the original equations that the attrition coefficient is static over time.

The concept of reinforcements was also eventually added into the equations. When a certain level of own forces is reached, called a breakpoint, additional reinforcements are called for during the simulation. Heterogeneous Lanchester Equations were also created where each side has many weapon systems, all of which have certain effectiveness in killing the enemy. If each side has m weapon systems with n variations of personnel operating those weapons, one can see how this difficult system of differential equations has no closed form solution.

In order to inject some variability into the Lanchester Equations and give them a stochastic element, Stochastic Lanchester models were created. These models depended upon utilizing a probability to determine whether the next entity killed would be from the blue or from the red side. Probabilities were determined by the ratio of the attrition coefficients of each army. While this did inject some variability into the use of these models, the problem of static attrition coefficients being utilized was still a valid one, since changes in force structure or morale had no bearing on the probability of which army's entity was destroyed next.

c. Force Ratio/Fire Power Scores

The idea of force ratio and fire power scores were to aggregate all the elements in a combat unit into one scalar measure of a force's combat effectiveness. The ratio of the two armies force power scores would then be calculated in order to determine the number of units destroyed in a time step as well as the forward or backward terrain movement of both the attacker and the defender. After the time step was complete, a new firepower score would be calculated based on the available units each force had remaining.

The firepower score is made up of the individual scores of the entities that make up that force. The values assigned to each individual unit are based on military judgment and actual measures of lethality. In this way real life testing and data of how effective certain weapon systems are against certain types of platforms are taken into account. The actual fire power score that is calculated each time step can also be altered for a variety of situations. The firepower score can be adjusted for different factors such as what type of battle the engagement is, battlefield terrain, morale issues, and logistics issues.

All of the above examples, with the exception of Stochastic Lanchester models, illustrate the use of Deterministic Models in simulation. The common factor they all share is the fact that once the initial inputs are known, the result is the same for all iterations of the simulation.

2. Event Driven Models

While deterministic models can give a baseline indication of how events will come to fruition, they lack variability. Decision makers desire to know the certainty of the decision they are making. With deterministic models the question of certainty is never posed. There is only the same exact answer every time. What is desired is some sort of simulation that gives an indication as to how close in ability and tenacity two different forces may be. It is important to know whether a conflict has a seventy five percent chance or a fifty one percent chance of success. If a simulation can yield two different results based on some sort of randomness built into it, a trend can be looked at over time and a success or failure rate can be determined. Event driven models yield an easy way to design this randomness into the problem. Event driven models basically utilize episodes called events that branch out into new events. Which events come next can be randomly or systematically chosen. This brand of simulation gave rise to a new type of decision-making aid.

a. Discrete-Event Simulation

A discrete-event simulation is one in which the system changes over time by a representation in which the state variables change instantaneously at separate points in time [Ref. 2:p. 6]. For example, a bank line simulation could have states enter the bank, get in line, start service, end service, and leave the bank. When a customer arrives, the enter the bank event occurs. In this case the get in line event occurs instantaneously. If no one is currently being served, the customer immediately goes to the service window. However, if there is already someone being served, the customer must wait in the line. Once the service window is clear, the next customer in line goes to the service window and the customer who was most recently served exits the bank.

In the above example randomness and variability can be injected into the problem by varying arrival times of customers as well as the service times. Different means and distributions can be used to vary the problem. Ultimately, the goal may be to determine what actual distributions occur in the bank with respect to waiting and service times so a decision maker can determine when it might be necessary to open an extra service window.

In addition to scheduling times, discrete event simulation can deal with the probability of which event will occur next in the simulation. For example, a simulation may contain a tank whose job it is to take out the main artillery battery. Once the state where the tank is in firing position is achieved, the next two states may be the tank hits the target or the tank misses entirely. If the tank hits the target, the next state may be to evade retaliatory fire. If it misses, the next state may be for the tank to stay in position and fire again.

By injecting variability into both which state an entity or simulation enters next as well as how long the simulation remains in that state, very different results can be achieved each simulation run. An analyst can look at success and failure over a long series of simulation runs to determine what the estimated success and failure rates for a situation may be.

B. USE OF MULTI-AGENT SYSTEMS FOR SIMULATION

While event driven models added another dimension to the use of simulation, their utility is somewhat limited to what the desires and intentions of the entities are prior to being put into the situation. In the bank situation, customers enter with the simple goal of going to the service window. Their actions will always reflect this goal and all their actions follow a script programmed into the simulation. This works well for simple cases like our bank example. However, for more complex situations, such as ground combat, scripts cannot be followed due to the enormity of the various emerging situations. The next step was to produce entities that write their own script based on a system of beliefs, desires, and intentions.

1. Multi-agent System Architectures

Multi-agent system architectures can vary widely from application to application. Some agent architectures utilize traditional Artificial Intelligence paradigms and others rely purely on numerical methods to create their agent behavior. Some basic forms of agent architecture are logic based, reactive, belief-desire-intention, and layered [Ref. 4]. The following sections describe these types of agent architectures.

a. Logic Based Agents

Logic based agents rely on the traditional approach to building artificially intelligent systems [Ref. 4]. The agents in this architecture create a symbolic representation of the environment around them and base their behaviors on this representation. The symbolic representation of the environment is usually held in some sort of database that the agent has access to. The agent’s database corresponds to “beliefs” in humans, that is, what the agent perceives the world around it to be. Based on the beliefs that an agent contains, it will run through a series of rules and, using logical deduction, base its next behavior on this set of known values.

The advantages to this type of architecture are that they bring a logical approach to problem solving. Anyone standing outside the simulation can deduce why the agent acted as it did because it followed a set of preprogrammed logical rules. There is a rationality that will always exist in the agent's decisions. Unfortunately, this type of architecture may be difficult to create because some environments do not lend themselves well to symbolic representation. Also, representing a dynamic, rapidly changing environment is difficult. Concrete time is needed for the agent to make a decision and the world around can quickly change before a logical decision can be made.

b. Reactive Agents

After pointing out the inadequacies with logic-based agents, researchers began to look at alternate designs. One of the early alternative designs was the reactive agent architecture. While many of these architectures have very different fundamentals, some of these basic themes occur [Ref. 4:p. 48]:

> - There is a rejection of symbolic representations, and of decision-making based on these representations.
>
> - Intelligent behavior is linked more to the environment that it occupies and is based upon the agent's interaction with that environment.
>
> - Intelligent behavior emerges from the collaboration of simple behaviors.

The architecture of reactive agents contains two basic defining characteristics [Ref. 4]. The first of these characteristics is that the agent's decision-making process is realized through a set action functions. In other words the agent constantly takes in data from its surrounding environment and maps one of its available actions to the situation. A simplified look at the basic rules that this architecture makes use of is of the form:

Situation -> action

For every situation, the agent attempts to match it with an action that will most benefit the agent's well being or goals.

The second defining characteristic is that many different actions can fire simultaneously. A certain situation in an agent's environment may be suitable for the implementation of more than one action, however, the agent must employ some sort of mechanism to select which action should be utilized. Typically, such agents may employ their own hierarchy. The agent may have actions divided into layers where lower levels represent a higher priority and inhibit lower priority actions.

It is important to note that the perception functions in these types of architectures rarely process or transform raw sensor input. There is usually assumed to be a pretty tight coupling between what the environment actually contains and what the agent perceives [Ref. 4:p. 49].

Reactive architecture in agents has many advantages. It is simple to implement and can provide some elegant solutions that create intelligent behavior in multi-agent systems. However, there are some drawbacks to this type of architecture. First, this architecture has a major dependence on what is perceived in the local environment at the present time. This limits the agent to a "short term" view as it maneuvers through its environment. Secondly, it is very difficult in a purely reactive architecture to create agents that learn from their environment and experience. It is not that this cannot be done, however, there has to be some change in the manner in which the agent maps an action to certain situations. Although this last point is a great way to create emergent behavior, it is also difficult to engineer agents that are suited for a specific task. Since there is no principled methodology, it is impossible to guarantee the behavior of agents in an environment over time and difficulty in discovering why the agent has adopted certain unwanted behaviors can become a daunting task.

c. Belief-Desire-Intention (BDI) Agents

Belief-Desire-Intention architectures rely on principles of practical reasoning, that is, the agent decides what it wants to do and then decides upon the best means of achieving it. In this architecture the agent decides moment by moment, which action to perform in order to further the agent's goals. The intentions of the agent and how it manages its intentions is very important in this architecture. For example, agents that hold on to intentions for too long may never be able to accomplish their goal or make full use of their abilities in other endeavors during a simulation because they just will not give up. For example, the minor league baseball player that holds on to his dream of playing major league baseball until he is forty years old illustrates this point. Alternately, the agent who constantly drops his intention when the first source of conflict arrives will never accomplish anything. The agent will keep drifting from intention to intention without letting any of them come to a successful end.

In order to create this architecture the agents require some sort of beliefs as well as a belief revision function. The agent must be able to alter its beliefs based on some sort of factual information that is being processed from the environment. Agents of this architecture also must be able to create a list of options of which actions are available to it at any moment given the agent's set of beliefs as well as its intentions. Lastly, as stated above, the agent must have some means of deliberating and choosing which intention in its list of options is the best one to pursue for the given moment.

One of the major advantages of BDI architectures is that they are intuitive. People make decisions in everyday life based on adopting a given intention and acting upon that intention until it is solved or it is given up. The second advantage is that it is easier to create agents for specific purposes. While this process is easier than implementing the same simulation with reactive agent architecture, there is still complexity, however, in ensuring that the agents created will pursue the behavior desired.

2. Examples of Multi-agent Simulation

a. *ISAAC*

In 1997, Dr. Andy Ilachinsky created his breakthrough simulation, ISAAC, which stands for Irreducible Semi-autonomous Adaptive Combat. Dr. Ilachinsky combined these four basic elements into a small unit called an ISAAC agent [Ref. 3]:

- A default local rule set describing how to deal with different situations
- Goals directing behavior
- Sensors generating an internal map of the environment
- An internal adaptive mechanism to alter the behavior or rules

By utilizing these four concepts, ISAAC was the first combat simulation to take advantage of the concept of artificial life.

ISAAC used a penalty function to determine the best position on a grid battlefield to move to next. It analyzed the results of moving in each direction and chose the best movement based on the result that best matched the goal of the ISAAC agent. The simulation also utilized personality traits. Personality traits determined the effect that each variable in the penalty function has on the overall result of the function. ISAAC also gave its personality traits a state dependent quality. An injured ISAAC agent's personality traits could be different than those exhibited when the agent was healthy.

b. Archimedes

The same concepts Ilachinsky used to create the ISAAC simulation have been used as the groundwork for the yet to be completed software called Archimedes.

Archimedes attempts to generalize the world of Multi-Agent systems. It can be used for military concepts, business problems, market concepts, or even political dynamics by breaking down agents' states into behaviors and actions, called variables and aspects.

Archimedes is one of the first attempts at utilizing a general architecture to create a variety of multi-agent systems for a variety of uses. The software has its own programming language as well as its own compiler that created the agents themselves. By doing this they allow the casual user who may not be familiar with programming languages such as C++ or Java to get quickly up to speed at creating multi-agent systems.

C. ATTEMPTS TO INJECT NON-HOMOGENEITY, VARIABILITY, AND ADAPTABILITY INTO MILITARY SIMULATIONS

1. Decision Making

In Pew and Mavor's discussion of military models, the authors criticized current military simulation as being too

> stereotypical, predictable, rigid, and doctrine limited, so it fails to provide a realistic characterization of the variability, flexibility, and adaptability by a single entity across many episodes. [Ref. 5:p. 151]

Their key argument was that humans are susceptible to aggressiveness, passivity, and even such things as confusion when making decisions. Current military simulations up until that point, however, had only provided decision-making processes that were based on doctrine and what a logical commander or soldier would do for a particular situation. In other words, military simulation failed to take human factors such as individual differences and human adaptability into their decision-making process.

Their paper outlined ways to inject these human factors into military simulation models. The two ways the authors proposed to do this was to take existing utility theories and modify them in certain ways that would allow individuals' variability, adaptability, and differences to be taken into consideration.

a. Utility Theory

Utility theory was a method proposed by Bernoulli at the beginning of the eighteenth century to improve upon the optimal method of making risky decisions by calculating the expected value. Utility theory allowed the decision maker to place his own attitude toward a decision's risk by creating a utility function that represented how risk favorable or risk averse the decision maker was. Utility theory over time ran into its own criticisms, however, mainly from an effect that fails to illustrate what decision makers tend to do called the "certainty effect".

The "certainty effect" can best be illustrated by the example of a roulette player who has just bet a thousand dollars on red and won, upping his total value to two thousand dollars. Utility theory, as it was proposed, stated that the same individual when faced with betting a large sum of money on a fifty-fifty gamble that yields double or nothing will use the same utility function when deciding whether or not to place a bet each time. However, humans have repeatedly shown not to act this way. Humans consider how negative results have been averted through chance in the past and tend to have the view that their luck cannot go forever; therefore they tend to take the route that will produce the most positive certainty after negative consequences have been beaten through the good graces of lady luck. In order to consider this aspect of human decision-making rank-dependent utility was proposed. Rank-dependent utility adjusts the utility function of a decision maker based on the cumulative probabilities the person is faced with.

Many times a decision maker is faced with conflicting utility. For example, maximizing enemy losses while minimizing own forces' losses is such a conundrum. While a commander may cause heavy enemy losses, he may also incur them as well. However, utility does not necessarily translate evenly across these two attributes. In other words, if a commander attacks, he or she may lose five soldiers while disabling ten of the enemies'. While the incurring of losses may have a low utility and eradicating ten enemy soldiers may have a high utility, this does not necessarily mean the commander should or will go forward with the attack. The utility from each conflicting attribute must have a way of translating from one to the other. This can be done using in an additive method or a multiplicative translation. In other words, using the multiplicative example, a commander may be twice as concerned with preserving his or her own forces as with eliminating the enemy's. Therefore, any utility gained from preserving one's own forces will be twice as valuable as the utility gained from disabling enemy forces. This method of utility theory is called multi-attribute utility.

b. Variability and Adaptability

The first thing recommended by Pew and Mavor was to ensure that simulation contained variability and adaptability, meaning decision makers in similar positions did not necessarily use the same method to determine what was the best course of action. Variability is necessary, they argued, to ensure that simulations provide some sort of unpredictability into the decision making process and to provide opportunities for decision-makers to learn and explore new alternatives other than the ones that are provided for through strict doctrine. Adaptability takes into account how a decision maker comes to view and reevaluate plans based on new and incoming information. They divided these models into three stages of development.

The first stage is random-utility models. By reformulating the utility as random variables, the constructor of the simulation can ensure that no two decision-makers place the same utility in a decision. While this method addresses the problem of making simulations more variable, it does not address the problem of ensuring the decision-making models are adaptive. In order to start addressing the problems involved in creating adaptive decision-making models, stage two or sequential sampling decision models were created. These models utilize a linear feedback system with stochastic inputs and a threshold output response function [Ref. 5]. This inhibitory threshold is used to determine when the decision maker has enough knowledge or indication that a decision can soundly be made. For example, important decisions would most likely have a higher inhibitory threshold than a decision that is unimportant. Thus, a decision maker is unlikely to act on important decisions until a reasonable amount of data has been collected that would justify soundly taking an action. The third stage models that Pew and Mavor refer to are Adaptive Planning Models. These models seek to allow the simulated decision maker to be able to look ahead as to how his present decision will affect future decisions. Since decision makers are capable of looking at future decisions that will come later and how the present decision may affect that future endeavor, these models provide an important piece of adaptive behavior.

c. Individual Differences

Decision makers differ in a lot of ways. Some decision makers may be very risk oriented while others may be very risk averse. In similar ways, decision makers can be optimistic or pessimistic or impulsive versus compulsive. The ways in which Pew and Mavor proposed introducing variability and adaptability into simulation allows for the insertion of individual differences in a simulation.

For example, impulsive decision-makers may have low inhibitory thresholds. That is, they tend to make a decision after even a small amount of information is received which favors choosing one action over another. Conversely, compulsive decision-makers would wait until a multitude of information that favors choosing one action over the other have been observed by the person making the decision. Another example of injecting differences among individuals is illustrated by changing the weighting factor in the multi-attribute model to favor taking out enemy forces rather than preserving one’s own if the leader making the decision is very aggressive vice passive.

Another factor that programmers constructing simulations need to consider is the state of the individual when making a decision. If the decision-maker is fatigued or frightened, does it affect how they make decisions? In the real world scenario it certainly does and should be reflected in military simulation.

2. Sensory Perception

Just as in the examples above for decision-making, it is important to take into consideration individual differences for such things as information processing, memory, and sensory perception. With the advent of agent driven simulations, such as ISAAC, it is important to account for how individuals perceive their environment through their natural senses and abilities. Soldiers walking through jungle terrain all have the chance of spotting an ambush ahead. However, individuals with the best eyesight, concentration, or even experience are the ones likely to first see the trap. There are individual differences among humans for even the most mundane of tasks such as recognizing objects that need to be considered if agent driven combat simulations, such as ISAAC and ARCHIMEDES are ever to forge ahead.

So is an agent in one of these simulations handicapped for the rest of their computer driven life if given poor ability to detect or sneak up on an enemy agent? Not if these simulations want to address what happens in the real world over time. Humans have not only the ability to improve, but also have the ingenuity to solve problems using other methods or even other senses to improve their performance in certain natural ability based tasks. For example, the basketball player whose feet are made of lead and has very little jumping ability may perfect the techniques of using pump fakes in order to get a shot off when closely guarded by a defender. Thus, it would be wrong to assume that an agent has the same ability in detecting an enemy in heavy cover six months into a campaign as it did during the first week when introduced to the environment.

If all agents can improve, the second question has to be how much can they be allowed to improve? Should all agents be able to reach the same proficiency in performing a certain task? Once again, if a reflection of how the real world works is desired then the answer is a resounding no. In the world in which we live, individuals are blessed or cursed with not just natural ability, but also with potential. While a certain individual may be terrible at a certain task, he or she may have the potential for very rapid and strong improvement. In a lot of cases, a person can only reach a certain level of improvement in a particular task or ability. There is a ceiling for all individuals at which little or no further improvement can be made. In human society, these ceilings are very varied and different among individuals. For example, Michael Jordan’s ceiling with respect to performing various tasks which lend themselves to successful playing of the game of basketball is much higher than maybe but a few others on the planet or even throughout history. Don’t get me wrong here. I do not mean to imply here that you’re typical NBA player did not do any work or preparation to succeed at the event in which they excel. However, there are individuals who could put in just as many long hours of practice or even more for that matter and still not achieve the same level of play that those who play in the NBA possess.

Another factor to consider when looking at raw abilities and the mastery of simple tasks in an environment is what specific factors instill ambiguity no matter what the skill or ability level of the person performing the task is. Going back to my basketball example, a general statement can be made that the closer a player is to the basket, the more likely he or she is to make the shot. While there are players who are better than others at making wide-open three point shots, almost always these players will still be better at making wide-open lay-ups only a few feet from the basket. It is important to identify these generalized maxims and employ them in your simulation.

D. USING COGNITIVE MODELS TO PREDICT HUMAN PERFORMANCE

David E. Kieras and David E. Meyer did some work at the University of Michigan that compared the actual performance of humans in doing a task with those of a cognitive model. Their goal was to create realistic human behavior and ability with respect to the manipulation of a software interface so that the model could be used in predicting human performance on other interface designs. The goal here is to reduce the cost levied by industry of having to actually create prototypes at every iteration in the design and have actual human subjects test them.

The authors specified that there were three elements necessary to construct and apply a model of human performance [Ref. 6]. The first of these elements is to actually have a specified architecture. In the case at the University of Michigan, they utilized EPIC as their underlying architecture. Secondly, a representation of the task strategy is required. This allows the architecture to directly correlate what it needs to take into consideration and how to perform the different tasks required. Thirdly, a method for identifying what strategy will be used in performing the task is required.

By gathering the three elements listed above and then running both the cognitive model and human subjects through the tasks required, it is possible to compare the results of both computer and human performance to ensure that the human results are accurately modeled by the cognitive architecture. As many iterations as required can then be accomplished until the cognitive model, does, in fact, represent human performance.

However, there is a caveat to all that has been described thus far. Just because a model can be programmed that actually fits human performance data, does not mean you have a useful model in predicting human performance [Ref. 6:p. 11]. Since this phenomenon is true, the authors came up with the concept of the *bracketing heuristic.* The idea behind the bracketing heuristic is that once you have identified the base strategy it is necessary to model the slowest reasonable execution of this strategy and also the fastest reasonable strategy of predicted human performance. By doing this you ensure that you have modeled both your alert and conscientious users and your slower and less interested users. The slowest versus fastest reasonable strategy does not just represent the user's interest or care about the task at hand. Other things to consider when working with human subjects are fatigue, ability, and other individual differences.

Once the bracketing heuristic has been done and you have baseline of data for both the slowest and fastest reasonable strategies, the designer can ensure that actual human performance data resides in between these two baselines for a variety of different trials.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DESIGN OF AGENT ARCHITECTURE

A. GENERAL ARCHITECTURE

In order to create agents that exhibit a modeled perceptual ability, it was necessary to create some sort of symbolic construction of the environment that is embedded within the agent. As was stated in the previous chapter under symbolic architectures, it is important to note that not all knowledge within an agent can be represented symbolically. However, for the purposes of my work, it was important to assume that the attributes that human senses are able to perceive are quantifiable and thus can be symbolically represented in some form or another.

The basic architecture I used to create my agents is the Symbolic Reactive Agent architecture [Ref. 15] and is shown below in Figure 1. As you can see, the basic components are the agent, the reactive agents that the agent contains and the symbolic constructors that the reactive agents have access to.

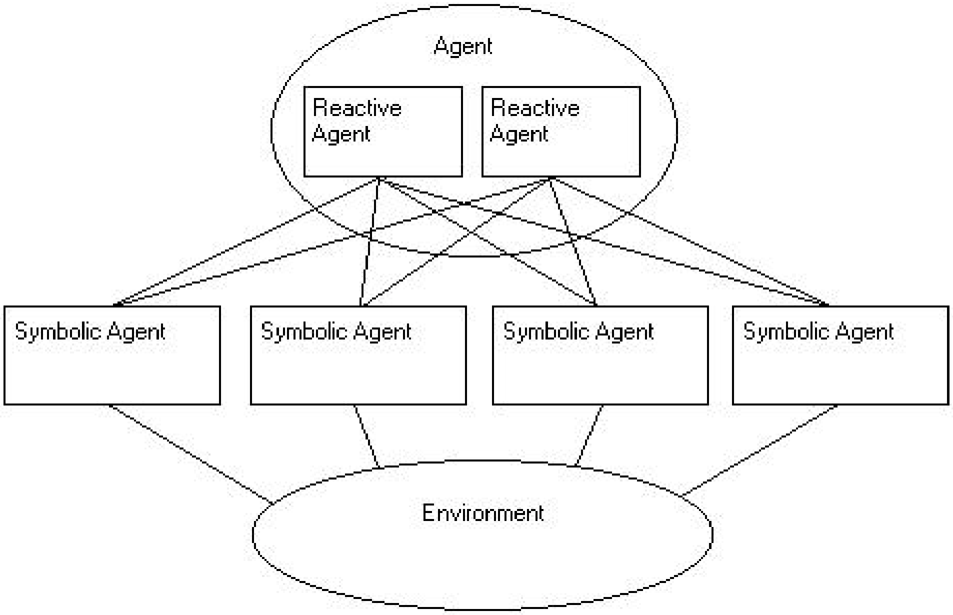

Figure 1. General Agent Architecture.

1. PathAgent

The main agent, or PathAgent, contains any number of Reactive Agents that it utilizes to make decisions. The Reactive Agents can be thought of as the agent's decision-making or logic component. The Reactive Agent gets data and parameters from each Symbolic Constructor that it has access to. This parameter or data received by the Reactive Agent is up to the designer as is the way he or she wishes to represent the environment symbolically.

The Reactive Agents within the PathAgent can be designed to give the agent either a numeric value or a String when the agent requests what it's recommendation for the next action within the agent's decision-making cycle is.

2. Reactive Agent

As stated above, the Reactive Agent is the agent's way of making a decision on what course of action to take next. The Reactive Agent takes into consideration the environment that has been constructed through its Symbolic Constructors to base that decision. Therefore, depending upon how accurate the symbolic environment within the symbolic constructor agent is, the information that the Reactive Agent is working with may or may not be accurate.

There is no formal way that the Reactive Agent informs the PathAgent what action it should undertake next. It is left to the designer as to whether they want to utilize something such as a polling type scheme where the next action taken reflects the majority of actions represented by String names returned to the agent through its decision-making function.

It should be noted that a Reactive Agent can have more than one Symbolic Constructor Agent that it uses to sample a certain parameter from the environment. In fact, by giving the Reactive Agent more than one Symbolic Constructor Agent, it is possible to increase the accuracy of the symbolic representation of the environment within the Reactive Agent.

3. Symbolic Constructor Agent

The Symbolic Constructor Agent's job is to sample parameters from its environment, store these parameters into its data storage Vector, and evaluate its present view of the environment after each sample of data sample is taken. The Symbolic Constructor Agent has a sampling rate that can be adjusted by the Reactive Agent that the Symbolic Constructor Agent belongs to. This way the Symbolic Constructor Agent can go between being very vigilant and very lax in constructing the symbolic representation that the Reactive Agent utilizes.

Not only can the sampling rate be utilized to help or hinder the Symbolic Constructor Agent's buildup of its environmental knowledge, the Symbolic Constructor Agent can be designed such that some sort of interference dilutes or alters the information from the environment that comes to the agent through the sample. For example, if an agent is trying to determine the size of an object to assess a classification, the sample from the environment will be altered so that, in general, the exact value the agent is trying to estimate will not necessarily be what the exact value from the environment is. The way the interference filters and changes the data is illustrated below in Figure 2.

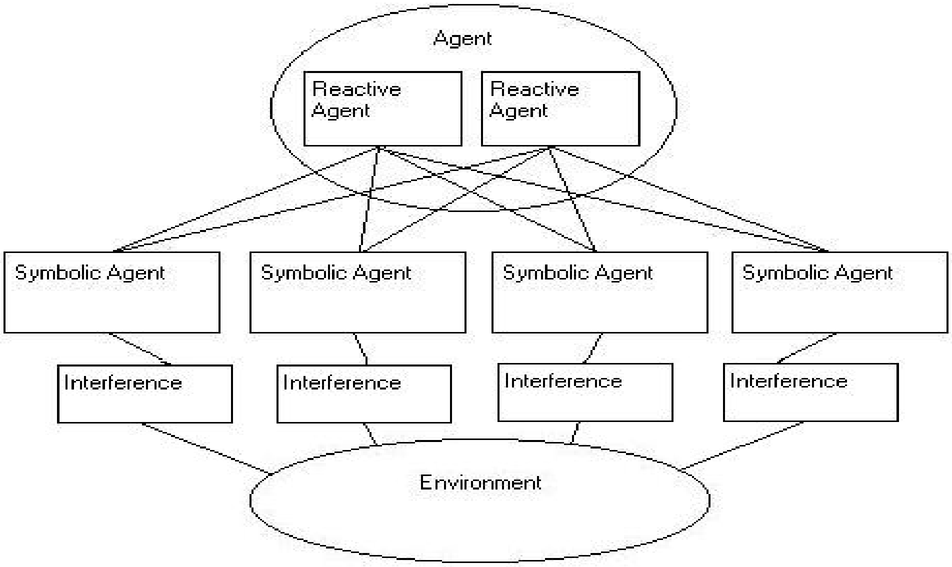

Figure 2. Agent architecture showing filtering and altering of data by the "interference" unit.

4. RCA and SCA Managers

The RCA Manager is what the agent utilizes to manage its various Reactive Agents. The RCA Manager is the class that allows the Agent to add, remove, store, and retrieve the various Reactive Agents that the Agent contains. The RCA Manager is a base class written in the Java programming language and is contained within the Agent base class.

The SCA Manager acts in the same capacity as the RCA Manager. It performs the same functions with respect to adding, retrieving, storing, and removing Symbolic Constructor Agents.

B. PERCEPTION AND SYMBOLIC CONSTRUCTION

As discussed in the previous section, the architecture proposed lends itself to modeling the way in which humans or any living creature for that matter take in and perceive the environment around them. As mentioned above in the architecture description, the agent’s Reactive Agents can each have one or even many Symbolic Constructor Agents. This model represents how human beings have certain resources that they can allocate to different lines of thought or concentration when it comes to thinking and perceiving the world around them.

For example, athletes often describe going “into a zone” where everything that they see or hear is within the game in which they are playing. Everything else they hear in the background, whether it’s crowd noise or stadium music, in their minds, is not even being consciously perceived. It’s not that their auditory senses are turned off, quite the contrary, if a teammate calls for the ball or the coach calls out a play, they can hear and process that stimulus even within the roar of the crowd. It’s just that they have allocated all their concentration to focus very narrowly on the task at hand. There’s still a threshold that can break this concentration, however. For example, if an explosion occurs, all people in the stadium, even those who are intently involved in playing the game, will be conscious of the noise.

The way to implement this perception and “fuzziness” in raw sensory data is the difficult part. As is the problem with symbolic construction within agent architectures, symbolic representation is limited to what can be built with a computer’s basic value types. This limitation means that the generalization of knowledge may have to be modeled by something other than the way this generalization is actually pictured in the mind of the person. For example, a person who has memorized the positions of buttons of different shapes spread out on a panel knows and comprehends their spatial relationship.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. DESIGN AND IMPLEMENTATION ISSUES

A. TASK SELECTION

The first step in designing the agent system was to actually select the task that is going to be modeled. The task needs to have both a perceptual task and a logical task with the logical task being dependent on how well the perceptual task is performed. The task needs to be simple enough to allow a decision maker to be able to perform the task if the perceptual symbolic construction within the agent is correct.

Constructing agents to do tasks based upon an internal symbolic construction of their environment is only limited by the speed and memory of the system one is working with. The number of data points and things the agent's perceptual knowledge can hold is only limited by the amount of memory one has on their computer. When working with humans in a real environment, however, the number of data points is limited by a variety of factors such as how long ago the information has been last retrieved and how important the memory was to begin with.

Secondly, the computer has the ability to perform rapid and deep-rooted arithmetic operations that humans cannot emulate. Despite the human's uniqueness and creativity in its thought process, this fact gives the advantage to the computer in problems of this nature. Given these basic differences in computers with respect to humans, it can make the design of realistic human behavior within an agent a difficult task.

Given these requirements, the task that was chosen was a distance estimation and path finding exercise in the NPS quad area.

1. Task Description

The distance estimation and path finding task consists of modeling a human who has both starting and stopping waypoints and has a set of intermediate waypoints that, while they have to hit all of them, can be hit in any order. Figure 3 below shows the map of the Naval Postgraduate School and the different waypoints of the positions that were mapped out.

The points in the quad that were chosen to be marked as waypoints were chosen so as to be as spatially balanced as possible. While most of the points have a physical object standing on them, some of the points do not. The list of the points is as follows:

H – Bench in front of Spanagel Hall

1 – Upper left corner of the garden in front of Spanagel Hall

2 – Lower left corner of the garden next to Root Hall

3 – The NPS clock

4 – Bench across the sidewalk from the “home” bench

5 – Bench in the middle of the quad

6 – Sea mine next to Bullard Hall

7 – Point at the base of the stairs in front of Spanagel

8 – Garbage can next to Bullard Hall

9 – Bench in front of Bullard Hall

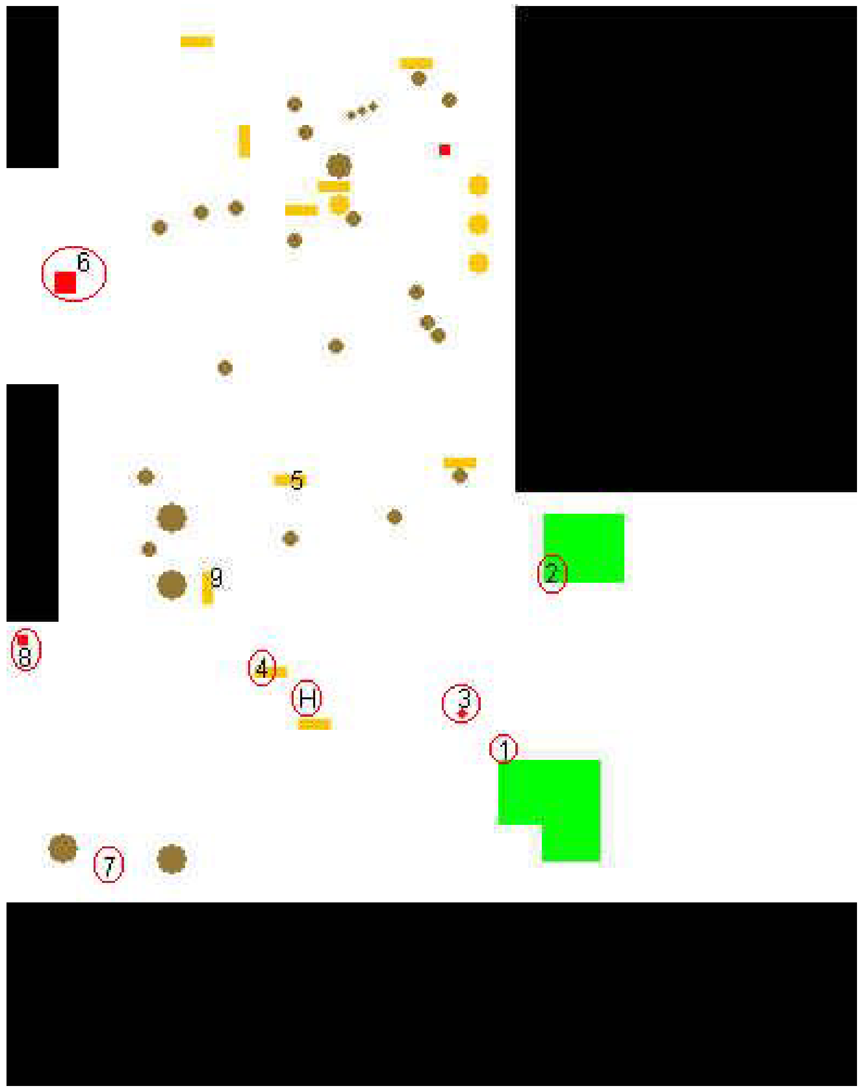

Figure 3. Map of the NPS quad and mapped waypoints.

In order to map the quad, I utilized a distance wheel and constructed a table that contained the distance from any given point to any other given point. Once that had been accomplished, I utilized graph paper to map the components themselves. After I was able to put a couple of points down, I utilized distances from at least two and sometimes three points to get the next point laid down by either putting it in the location where the lines bisected each other or where the lines triangulated a position. After the map on paper's distances matched those distances measured from the distance wheel, I recorded the horizontal and vertical positions of each point and constructed the figure shown above in Figure 3 using the standard Java2D package.

In order to design the study of how humans act on logical tasks that require action based upon imperfect knowledge of the surrounding environment, it was necessary to do a preliminary task analysis.

2. Task Analysis

By concentrating the breakdown of the task analysis into just the path-finding portion of the task, I posed that there are two different tasks the human or agent has to perform in order to choose the shortest path. First, the agent or human subject has to create a mental of model of where certain waypoints exist and how close or far they are from one another. Secondly, the subject has to go through some logical thought process to decide the actual path he or she will take. This thought process is obviously constrained by how well the subject has perceived the environment around them.

The task of creating the mental model is done differently if the subject is human vice one of the computer simulated agents. For this task, the human subjects have to rely on their sense of distance through both visual cues and the haptic feedback that one gets from walking a certain length. The agents, however, in an object-oriented language such as Java, get their distances from the symbolically constructed environment contained within the agent itself. All the agent has to do is query the position of the object from the object’s function which returns an exact x or y position, or the agent can query the simulation to ask how far it is from any of the other objects on the map.

B. AGENT SIMULATION

The agents were constructed by utilizing the architecture given in Chapter III. The agent could contain one or more symbolic constructor agents with which it constructs its internal environment. For this task, I decided that the agent portion would contain only one reactive agent that decided what course of action the agent would take based upon the rule that reactive agent contained. This way, it was possible for me to solely analyze the behavior of the agent when it has to act on imperfect perceptual information vice how the agent would weigh strategies to maximize its own performance.

1. The Task

The task chosen for this study is a problem that asks the subject to find the shortest route while still visiting a list of waypoints in the NPS quad that are represented in Figure 3. Point H was chosen as the home, or starting point, for the commencement of the simulation. Waypoints for the exercise are 1,2,3,4,7, and 8 of Figure 3. Lastly, the waypoint where the subject has to end up is at the sea mine at point 6.

There are several reasons why I chose these points. The starting point was chosen because it is very much centered in the playing field and all other waypoints are able to be seen from this vantage point. I felt that this was important given that the subject has to construct a mental picture using their visual senses from the very beginning. The waypoints were chosen because they spatially even out the area. This forces the subject to logically discern and decide what area of the quad they are going to go to first and how they are going to make a smooth transition from area to area before going to the last point. I chose to drop some points from the ones that I had originally mapped out to allow the human subjects to not be overwhelmed by the sheer numbers of how many routes that they can take.

Lastly, the last point was chosen because it is farthest from the home point and because it forces the subject to form a strategy to end up at this very remote point. Also, ensuring that the point of conclusion was different from the starting point, it prevented the subject from employing the circular strategy where they just try to go in the closest thing to a circle around the various waypoints. Strategies like I talked about above would have to be employed in order to ensure that the individuals made smooth transitions that didn’t waste a lot of ground when going from area to area while still being constrained to a distant endpoint.

2. Perception Types

In order for the agents in the simulation to correctly reflect the fact that humans do not get absolutely correct information from their senses when they are constructing their spatial representation of the locations of the different waypoints they must visit, it is important to select some sort of mechanism that provides the interference mechanism that was described in Chapter III. This interference mechanism ensures that the information coming to the agent is not the exact x or y position of the object it is querying.

When an agent queries the object for its location, the Symbolic Constructor adds or subtracts a value to the return of the function call depending on what type of agent the symbolic constructor is. The symbolic constructor agents contain a Java Vector type data structure for each waypoint and evaluate the data in this Vector by taking the mean after every sample and updating its overall x and y position for each waypoint.

a. Absolute Known

The first of these perception types is referred to as absolute known. This interference type does not add nor subtract any value from the exact x and y position returned to the Symbolic Constructor Agent. Thus, the agent will always have perfect knowledge of the relative distances between each waypoint. This perception type is more for the purpose of creating a reference situation as it obviously does not reflect the imperfect perception that a human subject will have when doing the very same task.

b. Ability Based

The second perception type is referred to as the ‘Ability Based’ perception. This type of interference models how humans have a certain ability to do certain basic tasks. The ability of the agent is represented by a parameter known as the agent’s ability score. This is a number between 1 and 100 and for the purposes of this simulation starts at 60. As the agent performs the task of estimating how far the distance to a certain point is, the ability score is incremented up. The agent continues to increment the ability as it samples from its environment until its score has reached a threshold that is selected at the beginning of the simulation by the user. The selection of this threshold is done by a toggle bar and is shown in Figure 4.

The threshold that the agent cannot overcome represents the maximum potential that this agent can achieve. Just like in nature, some subjects will be better than others and will reflect this in how high their ability score can go. This ability score is directly proportional to the interference that is added to the sample the symbolic constructor gets from its environment. Equation 2 shows how the ability score is implemented within the symbolic constructor agent to give a result that is proportional to the value of score. In other words, the lower the score, the more interference is added into the sample.

Equation 2:

Interference = (random integer % (maxInterference – ability)) X tag

Where:

Random Integer – A random integer generated by the Java random number generator

maxInterference – has the value of 100

ability – has a value from 60 to 95. This improves over time

tag – this value is a positive or negative 1. The Java random number generator also randomly generates this.

As can be seen in Equation 2, the value of ability can only reach 95. This insures that no matter how gifted or experienced that the agent is, there will never be absolute perfection in the values that the agent retrieves from the waypoint object. Thus, the agent will never have perfect information about the locations of the waypoints. The Ability Based perception mechanism is not based on any human performance data. Instead, it is more of a way to inject variability into the agent's perception mechanism. Some human perception tasks can be done perfectly once enough knowledge or experience has been achieved. This model seeks to represent these phenomena.

c. Ability and Distance Based

The third perception type utilizes a similar method as the Ability Based one with the exception that the tag listed above does not equal 1 or –1 but instead is the value of the distance away from the object the agent is divided by ten. This method models the accuracy with respect to predicting the position of the object to be more or less accurate depending upon whether the waypoint is far or close to the agent. In other words, if the object is far away, there is a lot of error built into the sample, but if the object is close this error becomes much smaller.

3. Decision Making Type

As stated in the task analysis section, the agent not only needs the ability to perceive and map out its internal environment, but it needs a strategy on which to achieve its goal based on the knowledge it has. In selecting the types of decision-making models to use for the Reactive Agents, I remembered the bracketing heuristic discussed earlier in Chapter 2. In order to try to put together a good human model, I tried to first create a model that would reflect a highly intelligent and focused human and then create a model that would reflect the person who isn't highly skilled or focused on the task at hand.

a. Overall Route

In order to model the subject who takes the task very seriously and with a high degree of concentration, I chose a method called the *Overall Route* method. This method has the agent cycle through every possible combination of order of waypoints that the agent can possibly choose, assess the total distance traveled based on where the agent thinks all the remaining waypoints are relative to one another, and choose the path that results in the shortest distance.

Once the agent has arrived at the next point it has chosen, the agent repeats the process over again for the remaining waypoints. This way the agent not only continues to observe and update the perceived environment around it, but it acts on this new information once the next point has been reached.

b. Nearest Point

The method that represents the simplest way a person might choose the shortest path, other than random picking of the next point, is the method *Nearest Point.*

On the execution of this method, the agent selects as its next waypoint the nearest perceived waypoint it has left to visit. Like the Overall Route method described above, as the agent proceeds to its next point, it continues to sample and reevaluate its surrounding environment. Once the agent has arrived at its next point it chooses the closest waypoint based on its new evaluated information.

c. Nearest Three

This method is similar to the Overall Route method described above with the exception that instead of cycling through all possible routes, the agent looks at all the combinations of orders up to three points. This method was chosen because asking actual human subjects to cycle through more than three points mentally is hypothesized to be an extremely difficult task. For example, if the agent starts the simulation having to consider six waypoints, this alone makes him cycle through 720 combinations alone. The Nearest Three method takes into consideration the limits of processing power a human being possesses.

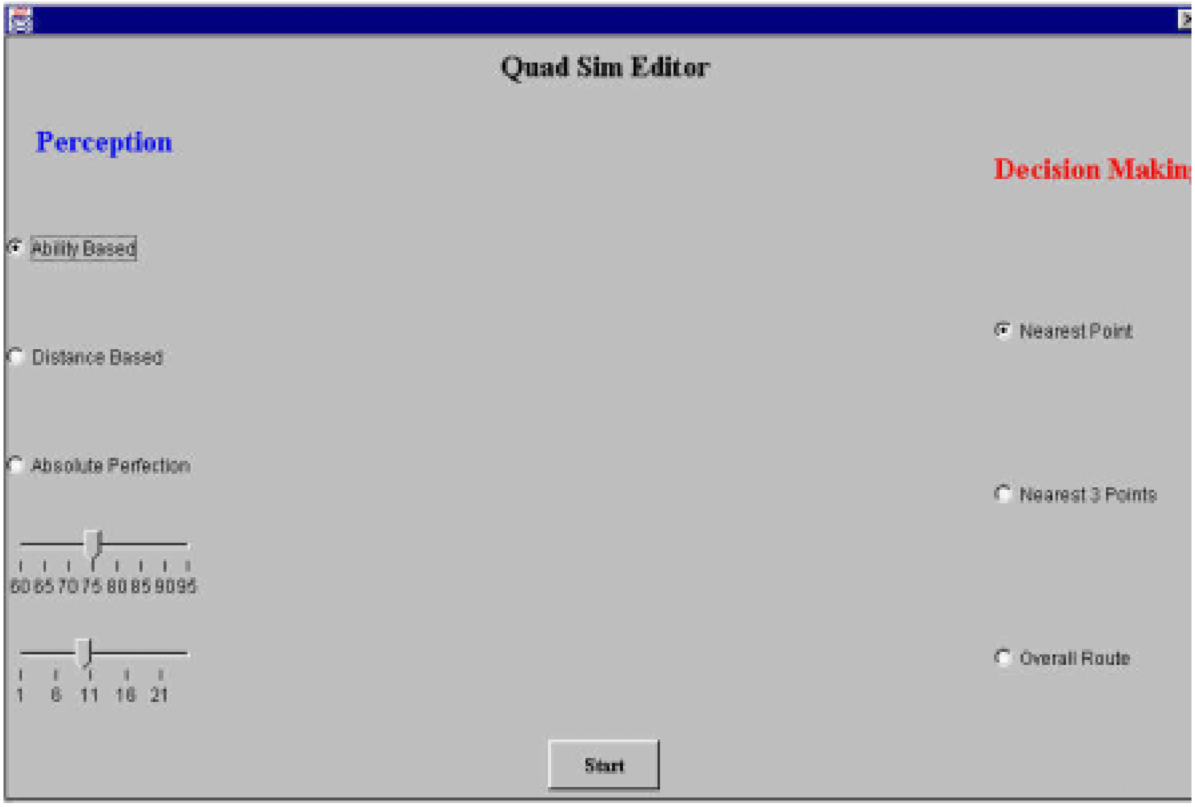

Figure 4. The Quad Sim Editor interface. This allows the user to select the combination of Perception and Decision Making type.

4. Simulation Combinations

In order to try and assess which types of perception and strategy combinations will translate into realistic human behavior, a series of trials for different combinations of perception and decision-making types is necessary. The list of the different combinations is listed below in Table 1. Table 1 shows the combination number along with which type of perception and decision-making model is chosen for the series of agent runs. Note that although the simulation allows more than one Symbolic Constructor agent to average in its data in order to construct the known locations of the different waypoints, I chose to keep the number of Symbolic Constructor agents at one. I did this because there is already a variety of combinations and my interest is geared more towards analyzing what types of interference models translate into the same uncertainty that human beings have when carrying out a similar task.

The PathAgents have a sampling rate at which they sample data points from their environment. The time cycle of the agent is based on clock cycles and the agent can be programmed to sample from his environment every certain number of these cycles. For the first data runs, I set the sampling rate at 5, or every fifth cycle. This sampling rate allowed the agent to quickly form too accurate a picture of the environment mainly because of the sheer number of samples that are taken. In light of this, I tried setting the sampling rate at 100. These runs had a consistent ability score of 60, an Ability Based perception, and cycled through the different types of Decision Making types.

Combination	**Perception Type**	**Decision Making Type**	**Ability Setting**	**Sampling Rate**	**Number of Total Runs**
1	Ability Based	Nearest Point	15 runs each at 60, 70, 80	5	45
2	Ability Based	Nearest Three	15 runs each at 60, 70, 80	5	45
3	Ability Based	Overall Route	15 runs each at 60, 70, 80	5	45
4	Ability and Distance Based	Nearest Point	15 runs each at 60, 95	5	30
5	Ability and Distance Based	Nearest Three	15 runs each at 60, 95	5	30
6	Ability and Distance Based	Overall Route	15 runs each at 60, 95	5	30
7	Absolute Known	Nearest Three	NA		1
8	Absolute Known	Overall Route	NA		1
9	Ability Based	Nearest Point	60	100	15
10	Ability Based	Nearest Three	60	100	15
11	Ability Based	Overall Route	60	100	15

Table 1. Perception type, decision-making type, and ability score for each of the agent runs.

At the beginning of the simulation run, the agent performs an analysis phase where all it does is sample the locations of the different waypoints at the sampling rate that has been specified. This phase is to reflect the way a human subject might try and construct its inner model of the environment that the problem is being performed within. Before starting on the simulation run, the agent writes the distances to each waypoint to a file.

C. HUMAN PERFORMANCE STUDY

For the human subject portion of the shortest path finding task, the task itself was no different than the one performed by the agent in the Quad Sim program. The human portion of the study was broken up into two phases. The first phase was the distance estimation portion where the human subjects estimated the distance from the various waypoints to where the subject was standing at the home bench. The second portion of the task involved the human subject attempting to choose the shortest path that was possible just as the agent did in the Quad Sim program.

1. Phase 1

During phase one the subjects were informed that they were going to give the proctor their best estimate of the distance from where they are standing at the home bench to each of the various waypoints in feet. The subjects were instructed that at any time they could be given what any of their previous estimates were. This way if they wanted to use one of their previous answers as a reference to future ones, they could use this information. The subjects were then asked how far they thought they were from all of the waypoints that would be utilized in the shortest path phase of the study. All subjects were asked for the best estimate of the various waypoints one at a time in the same order for all subjects.

2. Phase 2

The second phase of the exercise consisted of informing the participant that they would be trying to find the shortest path while starting at the home bench, ending at the mine waypoint, and hitting all the intermediate waypoints in between. It was emphasized to the participant that their goal was to consider the straight-line distance between points when trying to construct the shortest path while still hitting all the waypoints. How far the subject had to change his or her path because of obstructions was not to be taken into consideration.

The subjects were allowed to question the proctor as to what remaining waypoints they still had to visit and take into consideration at any time during the task. Subjects were also informed that the way they were to proceed from waypoint to waypoint was to create their plan of action and then announce to the proctor which waypoint was the next one they were going to visit. Once participants had announced their next waypoint and started walking towards it they were not allowed to change their mind and select a different waypoint in mid-stride. This was done to reflect that the agent in the computer simulation was constrained to walk towards the waypoint it had chosen and could not switch from that intention.

The subjects performed the exercise until they determined the correct path that minimized the overall distance of travel and still visited all the waypoints or until they had performed three trials, whichever event came first. The participants were required to walk the whole path until the last trial had been pre-determined so as to try and familiarize them with the overall layout of the trial area. At the end of each trial, the participant was asked to provide a short explanation as to what their strategy for choosing the path they had just completed was.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND CONCLUSIONS

A. HUMAN AND AGENT COMPARISON

The first analysis on the data was a human versus agent comparison for the path-finding task. The questions that I wanted to ask with this comparison are how do humans compare with their agent counterparts regarding variability of routes taken and the raw ability to do the task at hand. As I stated in the previous chapter, many different agent combinations were used to get a baseline of data to compare with the human subjects performance. My first analysis was to find the average distance traveled for each trial for both the human and agent subjects.

1. General Comparison

Figures 5, 6, and 7 show the results for each combination of agent that utilized the Ability Based interference mechanism in its Symbolic Constructor agent

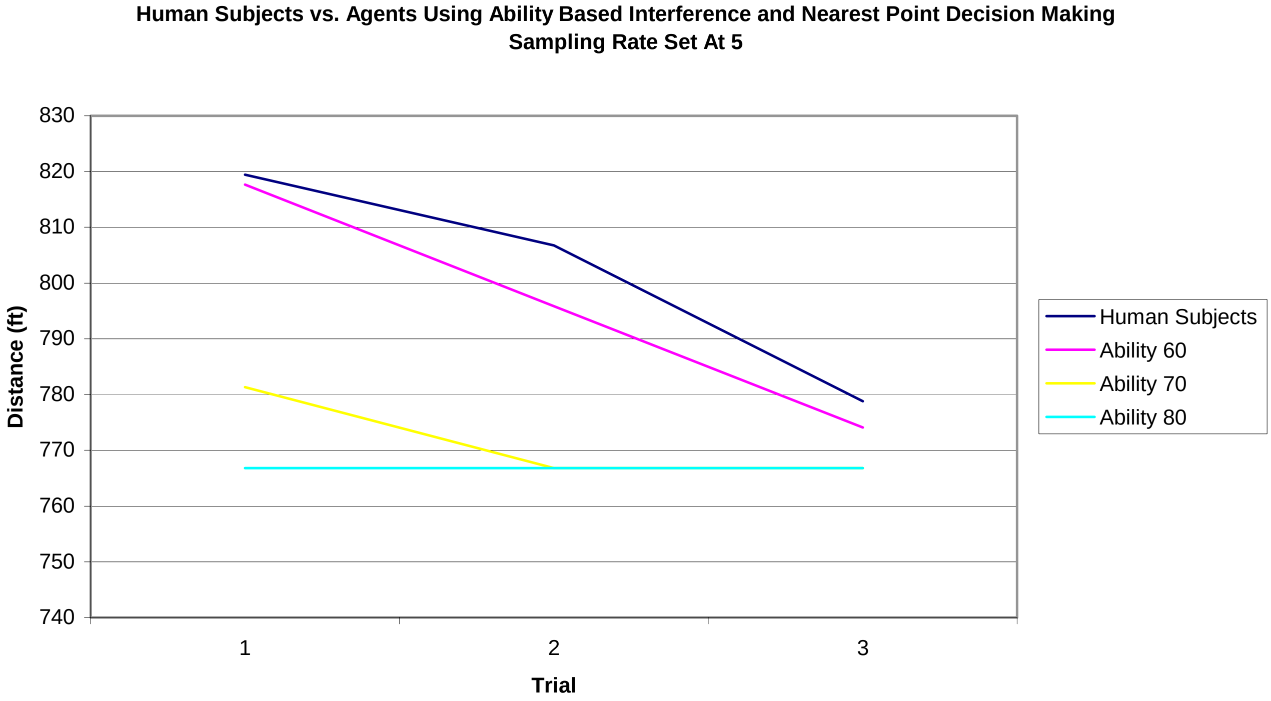

Figure 5. Human Subjects against Ability Based Interference Agents utilizing the Nearest Point Decision Making Type for three different ability scores.

All the agents in these figures had a sampling rate of 5 and all three ability scores are represented for the given Perception and Decision Making type.

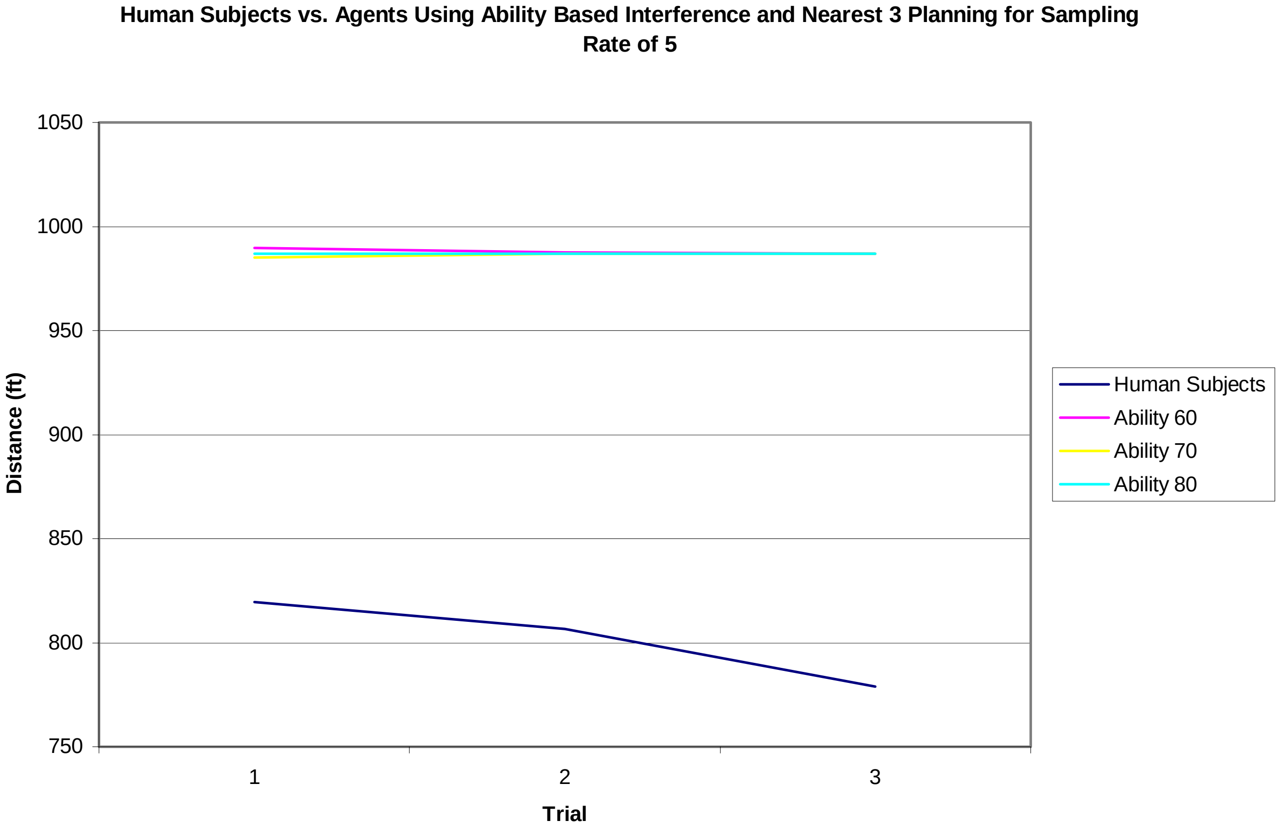

Figure 6. Human Subjects against Ability Based Interference Agents utilizing the Nearest Three Decision Making Type for three different ability scores.

As can be seen from these graphs, the agents seem to quickly and accurately symbolically construct the relative positions of the waypoints within the environment when the sampling rate is set at five. With the exception of the agents with a Nearest Point decision-making mechanism with an ability score of 60 in Figure 5, the agents seem to quickly reach the optimum level of performance when the agent has a perfect knowledge of its environment no matter what the ability score is set at.

The Decision Making type has more to do with how the agent performed than did what the ability score was set at for these trials. Again, this seems to stem from the fact that the agents were all starting the task with near perfect information because of the large number of sample the agent had taken before starting the task. The Nearest 3 method shown in Figure 6 consistently yielded worse than human results. By looking at what the agent does with this decision-making mechanism with absolute knowledge, the agent in this scenario tended to concentrically work its way out from the home base. This seemed to be because the agent would first go to the bench at waypoint 4 first and then went to the right side. Once there the agent had two choices, it could go to the other two waypoints on the same side and then set itself up for a long leg to get to the other side, or it could go to the nearest one on the other side and then cross back to the closest waypoint on the other side from that one. Basically, the agent performed a large crossing pattern across the quad model. Given a different placement of waypoints for a different scenario, the result may not be this inefficient.

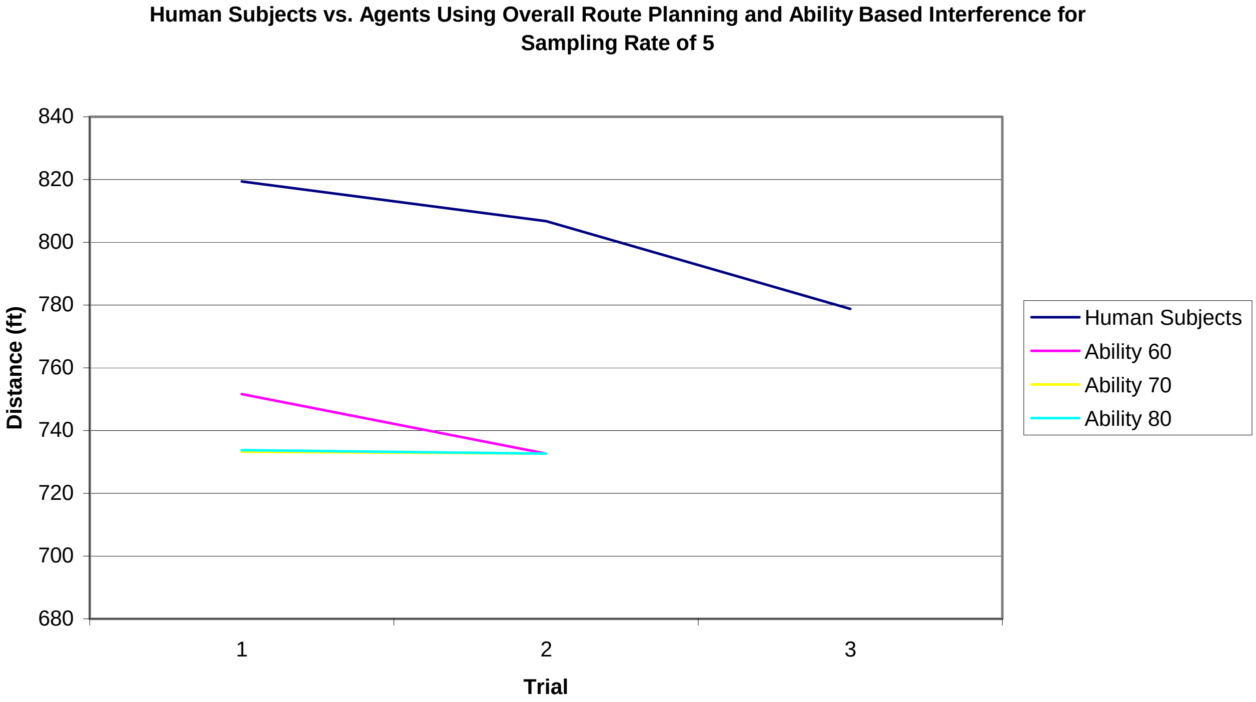

Figure 7. Human Subjects against Ability Based Interference Agents utilizing the Nearest Three Decision Making Type for three different ability scores.

Figures 8, 9, and 10 show similar results for the agents that utilized the Distance and Ability Based interference mechanism. The agents here seem to show more difference among the combinations that had an ability score of 60 or 95. A reason for this is probably that the magnitude of the error added to the x and y components during each sample is a degree higher than those used in the Ability Based interference types. Since more chaos and error is injected to the agent's symbolic construction of its environment, this yields more variability in the performance of the agent.

Like the last examples, the Nearest Point Decision Making type seems to most reflect the performance of the human subjects. This time the Nearest Three Decision Making type had more variability, was closer to the human subjects' performance and also displayed another interesting characteristic. Because the inner symbolic environment of the agent is more chaotic and even wrong, the agent does as well or even better during trial one than at the end of trial three.

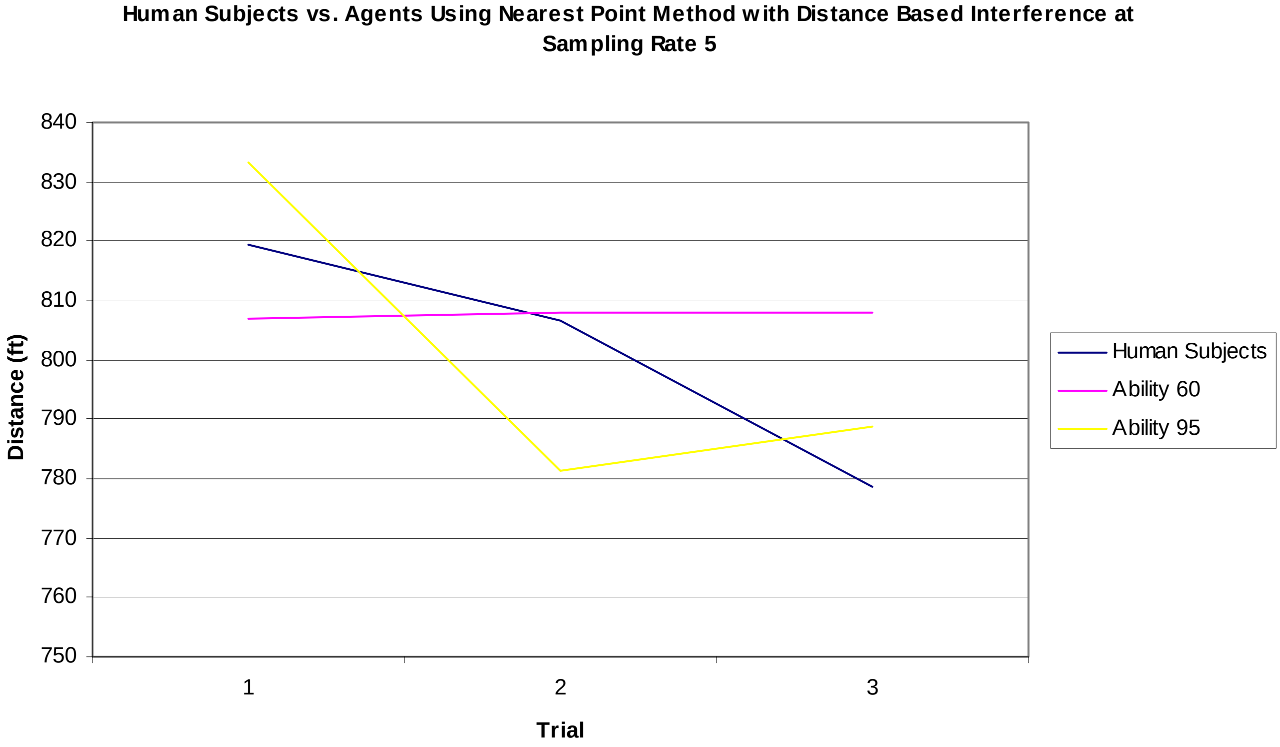

Figure 8. A plot of human Subjects against Distance Based Interference Agents utilizing the Nearest Point Decision Making Type for two different ability scores.

The results of the Overall Route planning Decision Making type agents that utilize the Distance Based interference mechanism exhibited in Figure 9 are very similar to those for the Ability Based agents utilizing the same Decision Making mechanism that is shown in Figure 6. The path planning mechanism inherent in this Decision Making type already has an extreme advantage over humans since perfect knowledge of waypoint positions will always yield a perfect result. The agent cycling through all possible routes seems to carry its inherent advantage over human subjects no matter how much interference is put into the agent's inner environment.

2. Statistical Significance

Another question that has to be posed is whether any of the combinations have the same statistical significance as the human subjects. In other words, if the agents in the virtual world are ever going to be said to act like humans, at least one or more of the combinations must exhibit nearly the same performance statistically as those of human beings. This way if an agent has some sort of internal monitoring that chooses the best perception method and decision making method to determine what is the best path to take for this particular exercise then this method will eventually push its way to the forefront.

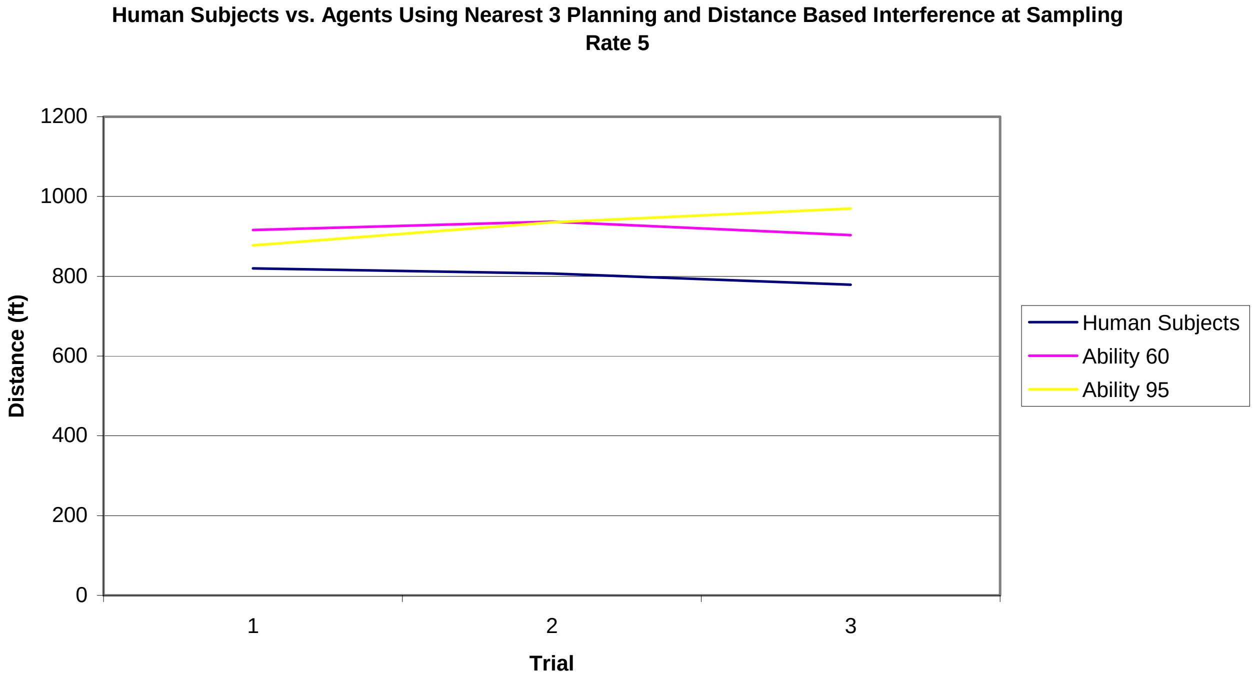

Figure 9. A plot of human Subjects against Distance Based Interference Agents utilizing the Nearest Three Decision Making Type for two different ability scores.

For each agent combination an F test ANOVA (p = 0.05) was performed between its overall mean for all trial attempts and those of the human subjects. Table 2 lists the results of this analysis. Note that all categories of acceptance for this particular task have the commonality of using the Nearest Point method to choose the way in which the agent solves the problem of selecting the shortest overall distance. It should also be noted that in order for the Nearest Point type agents to statistically reflect real-life human subjects a certain amount of interference has to be built in. Both ability based agents that utilized the Nearest Point decision-making technique had ability scores of 60. For the distance based agents that utilized the Nearest Point method ability, scores of both 60 and 95 yielded human-like performance. This can probably be attributed to the fact that the distance based interference mechanism puts more pure interference into the inner environment because of its distance driven multiplicative factor. Thus, enough interference is generated to give the agent similar performance as those agents that utilize the Ability Based interference technique with the ability score very low.

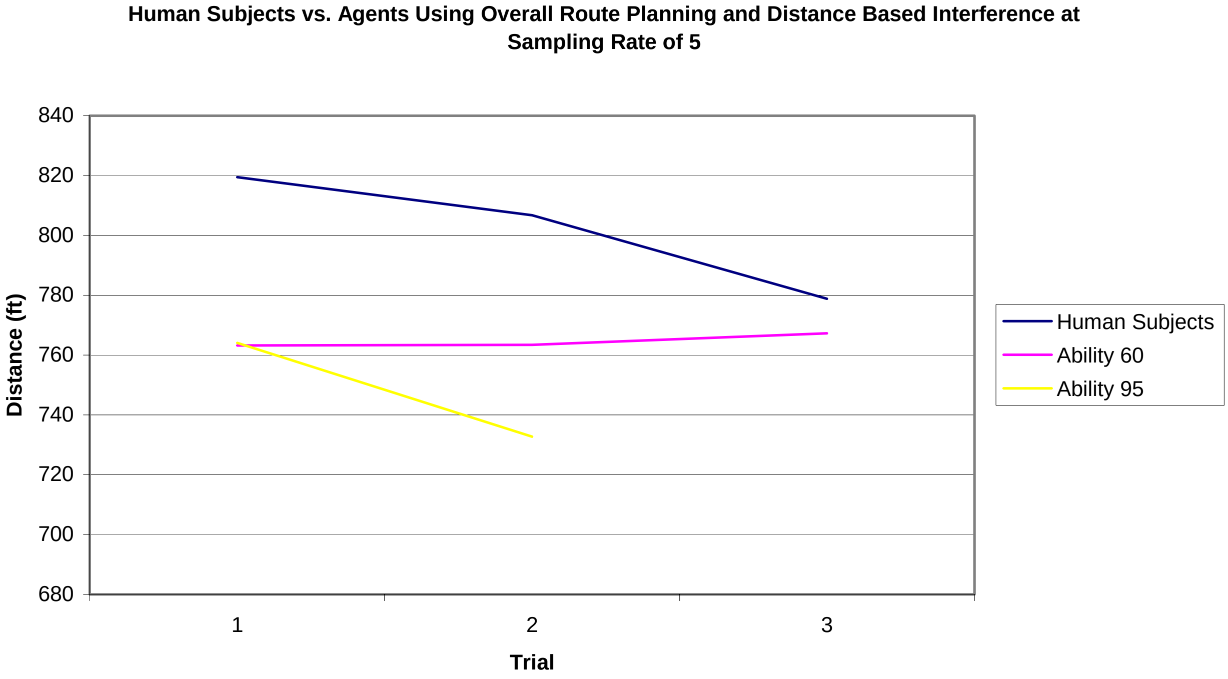

Figure 10. A plot of human Subjects against Distance Based Interference Agents utilizing the Overall Route Decision Making Type for two different ability scores.

I emphasize once again that just because these perception and decision-making combinations were found to statistically be the same as human performance for this particular task, this does not mean this result will transfer across similar tasks. A different task may require a completely different strategy to be implemented and the results may be very different if a different placement of waypoints is used. More will be covered on this subject later in this chapter in the discussion of the Bracketing Heuristic.

3. Qualitative Results

Type	*f*	F	Accept/Reject
Ability Based, Nearest Point, ability score 60, SR 5	0.33	4.0	Accept
Ability Based, Nearest Point, ability score 70, SR 5	14.38	4.0	Reject
Ability Based, Nearest Point, ability score 80, SR 5	26.42	4.0	Reject
Ability Based, Nearest Three, ability score 60, SR 5	712.67	4.0	Reject
Ability Based, Nearest Three, ability score 70, SR 5	696.15	4.0	Reject
Ability Based, Nearest Three, ability score 80, SR 5	709.30	4.0	Reject
Ability Based, Overall Route, ability score 60, SR 5	20.67	4.0	Reject
Ability Based, Overall Route, ability score 70, SR 5	36.45	4.0	Reject
Ability Based, Overall Route, ability score 80, SR 5	38.21	4.0	Reject
Distance Based, Nearest Point, ability score 60, SR 5	0.19	4.0	Accept
Distance Based, Nearest Point, ability score 95, SR 5	0.01	4.0	Accept
Distance Based, Nearest Three, ability score 60, SR 5	28.73	4.0	Reject
Distance Based, Nearest Three, ability score 95, SR 5	45.69	4.0	Reject
Distance Based, Overall Route, ability score 60, SR 5	12.00	3.96	Reject
Distance Based, Overall Route, ability score 95, SR 5	19.39	3.96	Reject
Ability Based, Nearest Point, ability score 60, SR 100	0.05	4.0	Accept
Ability Based, Nearest Three, ability score 60, SR 100	304.22	4.0	Reject
Ability Based, Overall Route, ability score 60, SR 100	14.49	4.0	Reject

Table 2. F Test ANOVA (p = 0.05) analysis between average human subject path distance and all average agent combinations

While it is true that certain types of agent combinations were statistically the same as human performance, there was not nearly as much variability exhibited by the agents as there was with their human counterparts. The agents tended to cycle through the same set of, in most cases, two to six paths. For the most part, these paths never deviated too terribly from best-case scenario given perfect knowledge for the given decision-making type. There was never the random detrimental move that human subjects seem to make when are on the right track towards correctly solving the problem.

Strategies that the human subjects used to determine the overall shortest path were mainly sectional in nature. Many subjects described dividing the quad into halves or sections and then trying to hit each waypoint in a section while setting up the transition to the next section of the quad. Another strategy mentioned was the avoidance of backtracking. In other words, subjects still divided the quad up into at least two sections and then avoided choosing routes that would cross the boundaries of the sections too often.

B. AGENT VERSUS AGENT COMPARISON

The second comparison of data that was performed was to look at the various data between agent subjects to determine which variables had significant effects on overall agent performance. The first comparison was to differ the sampling rates of the agent's symbolic constructor agents. Since the agent's symbolic constructor agents can sample from the environment at different rates, would sampling more or less frequently affect the results of the task? The second comparison was among agents of different perception mechanisms. It is obvious that different decision-making types will greatly affect the performance of the agent. Does varying the interference mechanism also have an effect on how well the agent performs the task?

1. Differences in Agents with Different Sampling Rates

As was stated in the previous chapter, there was a group of simulation runs done with agents that used a sampling rate of 100 vice the sampling rate of 5 used in the other runs. Figure 10 shows the comparison of the Ability Based agents with the ability score set at 60 for agents with a sampling rate of 5 and others with a sampling rate of 100 all other things being consistent.

Notice that from Figure 11 that agents that utilized different sampling rates yet used the same decision-making type had slight differences. In general, the agents with the higher sampling rate (sample rate 5) did a slightly better job of finding the shortest path than those agents that used the less frequent sampling rate. The exception for this is once again agents that used the decision-making type Nearest Three. By injecting more interference into these agents, performance actually improved since it made the agent deviate from the awful result that using the Nearest Three type with perfect environmental information yields.

2. Differences Between Agents with Different Perception Types

The second area of analysis was determining the differences in perception types with respect to agent performance. The main question to be answered is, does one or the other perception mechanism more correctly model human perception for this task?

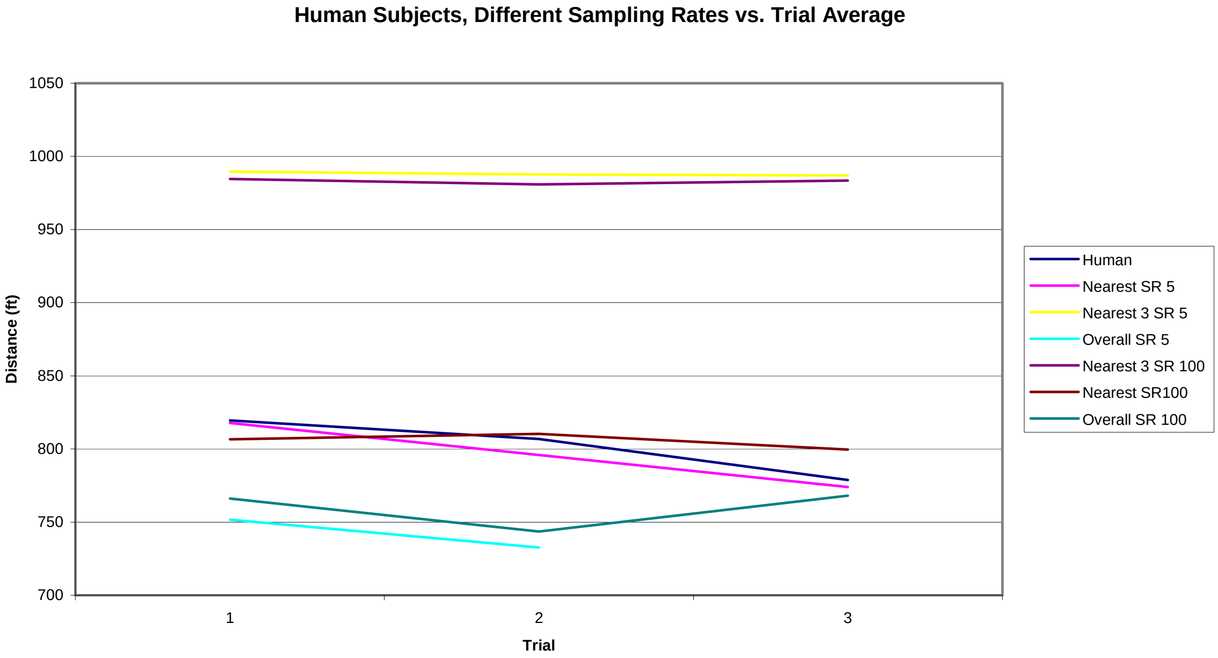

Figure 11. Comparison of agents with different sampling rates.

Basically the difference between the two types of interference mechanisms is that the Ability Based one applies a consistent error to each sample based upon its ability score whereas the Distance Based mechanism applies an error that changes depending on whether the agent is close to or far away from the waypoint in question. In most cases the Distance Based mechanism will also yield larger interference since the distance of the object only has to be above ten feet away in order for a multiplicative increase to be noted.

Figure 12 shows the results of comparing the two different types of interference mechanisms while holding the ability score at 60 and the sampling rate at 5 for all the different decision-making types. Like the pattern noted before, for the Nearest Point and Overall Route types, injecting more randomness and interference into the agent caused a slight decrease in performance. The Nearest Three type once again seemed to benefit from the less accurate inner representation of the environment.

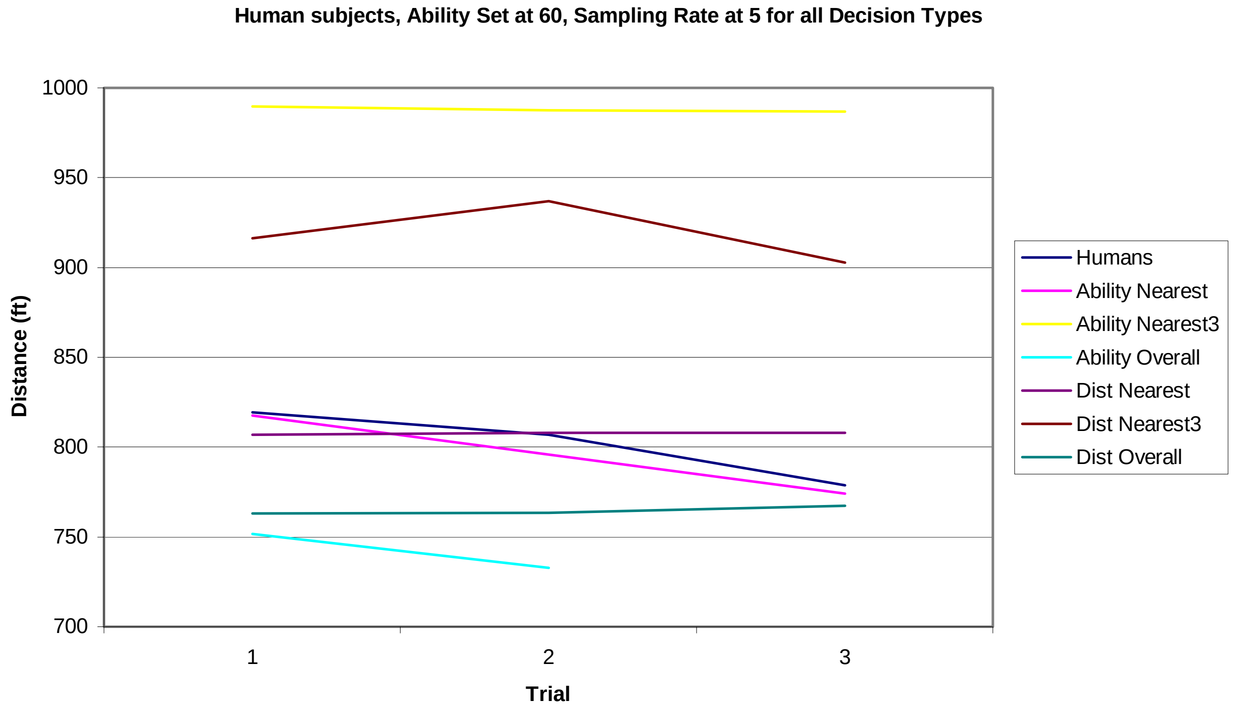

Figure 12. Comparison of agents with different perception types.

C. BRACKETING HEURISTIC

As I have noted previously, the use of the Bracketing Heuristic can help determine if a model will represent human behavior for a generalized group of tasks. It is important to recognize the difference between statistically representing human behavior for a specific task and creating a model that will represent general human behavior for a generalized set of tasks.

1. Statistical Significance

As stated in section A of this chapter, there were some combinations of agent types that yielded results that did not differ statistically from the performance of the human subjects. This means for this given task that these agent types can be used and the overall mean for the agents will be the same as human subjects performing this same task. If all that's desired in the simulation is to have the agent walk a path that is the same statistically as humans in real life, then using this combination is an acceptable solution. However, if the variability of real human behavior is required care should be taken with only putting these types of perception and decision-making mechanisms into the agent. As was stated previously, the agents did not venture down as many paths as their human counterparts, preferring to stick to the same safe group of paths that the decision-making algorithm yields.

2. Use of Architecture and Methods to Model Similar Tasks

In order to inject this variability into the agent, the user should probably put all the perception and decision-making types into an agent and let some sort of reward system allow the best methods for the particular environment the agent is in to come forward. In other words, effects such as shuffling the waypoints, changing the positions of the waypoints or changing the number of overall waypoints there are may affect which method or strategy is the best one to utilize for that environment.

Figure 13 below shows the results of all agent combinations and the human subjects (denoted by the dark gray line) for all three attempts. Note that there are some mechanisms that yielded better results than the human trials and some that yielded worse results. The result is that the human results are surrounded by the other results and "bracketed" in. This follows the concept of the bracketing heuristic and implies that a worthwhile task to determine if outfitting agents with these mechanisms for perception and decision-making would create an adaptive agent that is well suited to performing general path-finding tasks for a variety of different scenarios.

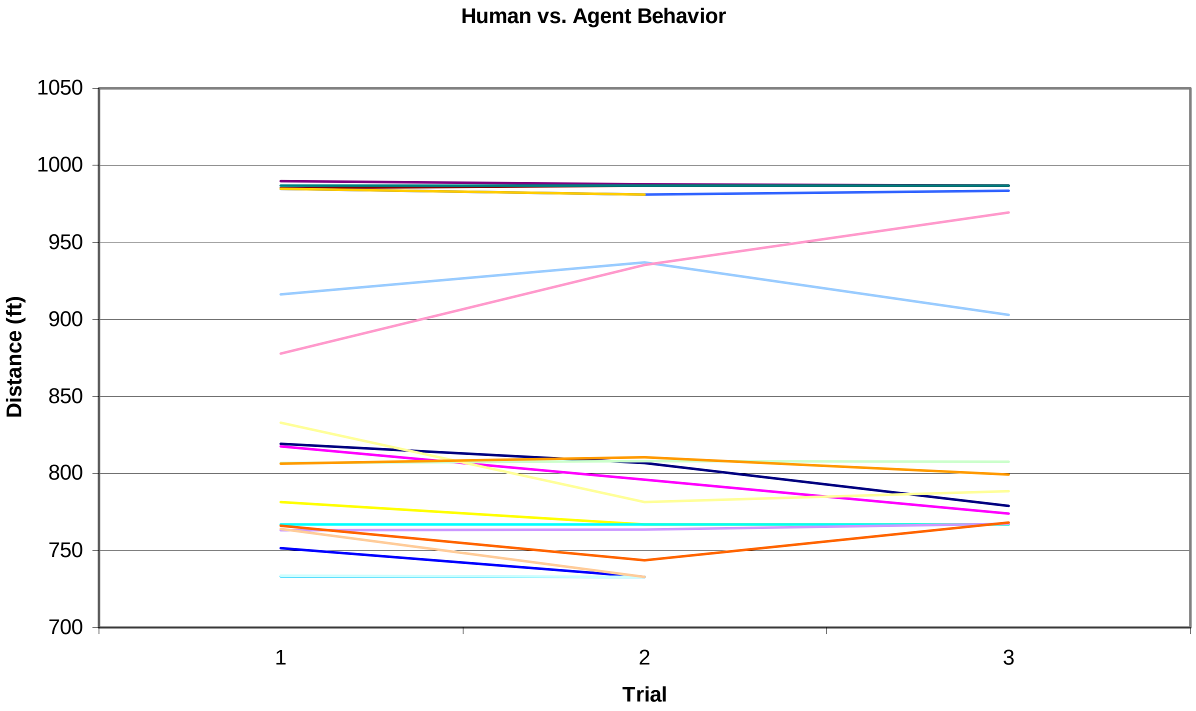

Figure 13. Graphs of all agent combinations and human subjects for all three trials.

D. OVERALL CONCLUSIONS

While some of the agent combinations were statistically the same as human behavior, the more important finding was that by analyzing all the different agent combinations with respect to human behavior, the potential for constructing agents that can adapt and use different techniques to solve a path-finding task may be possible. While the Nearest Point decision-making mechanism was the best and the Nearest Three method was the worst, which knows what the outcome might have been if different waypoints with different relative distances amongst each other were used.

Although the greatest variability among agents was due to its decision-making component, there were differences seen among different perception techniques within groups of agents that had the same decision-making component. Thus, the differences in how agents mapped out their inner environment yielded groups of agents that were better than others.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. RECOMMENDATIONS AND FUTURE WORK

The interference schemes used for the agents in this study used numerical values that varied from five to forty in the case of Ability Based agents and five to forty multiplied by the distance factor which in this case was distance from the object divided by ten. Given that the agent was probably at most three hundred feet from the waypoint, this means that the interference numbers could possible be as high as 1200 if the ability score was set to its lowest level. As I stated in the conclusions, the agent had to have a lot of error injected into its internal environment before degradation in agent performance is achieved. One question that still needs to be answered is does this variability in agent performance continue to decrease or increase depending on how large the interference is or does it possibly reach a saturation level.

The theory that this agent architecture is a good model for different path-finding situations because the bracketing heuristic applies over the performances of different agent combinations is another avenue worth exploring. Perhaps, constructing an agent that has the capability to utilize all these types of perception and decision-making models and letting it choose what combination the it wants to use based on the environment it perceives and past successes or failures would yield an agent that adapts and learns how to handle a multitude of different environments.

Since symbolic agents are limited by the requirement that it must be able to symbolically represent components of its environment, further research on what types of symbolic representation are more effective than others need to be pursued. In this thesis the symbolic representation was very simple. The agent used a best guess for the x and y location of the object so it could get a feel for how far away the object was. Other forms of symbolic representation are not nearly as simple, but perhaps there are numerical methods that can be used to compensate for this limitation.

The last offshoot of future work has to do with further creating generalized methods for comparing agent to human behavior. In addition to the work done here, future possibilities involve multiple related tasks that utilize adaptive agents to determine whether agents utilizing this architecture exhibit the same adaptability to tasks that humans do. Using the present thesis topic as an example, if different quad areas could be mapped out, it might be possible to compare human subjects as they would go from environment to environment and perform a path finding task with those of agents that have a set of rules or mechanisms with which to act on its inner environment. As the agents go through their environment, like human subjects, they would have to rapidly adapt to such factors as the placement of waypoints to determine what the best decision-making strategy may be for the particular environment that the agent is presently in.

The work done in this thesis is only a start in determining the best way to compare human and agent behavior. Many different approaches need to be looked at before a generalized method for ensuring that agent behavior truly reflects that of humans can be put together.

LIST OF REFERENCES

1. Dreyfus, H.L. and Dreyfus, S.E., *Mind Over Machine*, The Free Press, 1986.
2. Law, A.M. and Kelton, W.D., *Simulation Modeling and Analysis*, 3rd ed., McGraw-Hill, 2000.
3. Ilachinski, A., *Irreducible Semi-Autonomous Adaptive Combat (ISAAC): An Artificial-Life Approach to Land Warfare*, Center for Naval Analyses, 1997.
4. Weiss, G., *Multiagent Systems, A Modern Approach to Distributed Artificial Intelligence*, The MIT Press, 1999.
5. Pew, R. and Mavor, A., *Modeling Human and Organizational Behavior – Applications to Military Simulations*, National Academy Press, 1998.
6. Kieras, D.E. and Meyer, D.E., *The Role of Cognitive Task Analysis in the Application of Predictive Models, of Human Performance,* EPIC Report No. 11 (TR-98/ONR-EPIC-11), University of Michigan, 1998.
7. Deitel, H.M. and Deital, P.J., *Java How To Program,* 3rd ed., Prentice-Hall, 1999.
8. Dreyfus, H.L., *What Computers Can't Do*, Harper and Row, 1972.
9. Dennett, D.C., *Brainstorms*, Bradford Books, 1978.
10. Genesereth, M.R. and Nilsson, N., *Logical Foundations of Artificial Intelligence,* Morgan Kaufmann Publishers: San Mateo, CA, 1987.
11. Konolige, K., *A Deduction Model of Belief*, Pitman Publishing: London and Morgan Kaufmann Publishers: San Mateo, CA, 1986.
12. Russell, S. and Norvig, P., *Artificial Intelligence: A Modern Approach,* Prentice-Hall, 1995.
13. Rosenschein, S.J. and Kaelbling, L.P., A situated view of representation and control. In P. E. Agre and S. J. Rosenschein, editors, *Computational Theories of Interaction and Agency*, pages 515-540, The MIT Press: Cambridge, MA, 1996.
14. Devore, J.L., *Probability and Statistics for Engineering and the Sciences*, 4th ed., Duxbury Press, 1995.
15. Hiles, J., *Course Notes for MV-4015 Agent-based Autonomous Behavior for Simulations*, Winter, 2001, Naval Postgraduate School.

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, VA

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, CA

3. Prof. John Hiles
   Code MOVES
   Monterey, CA 93943
   jhiles@mindspring.com

4. Prof. Rudy Darken
   Code MOVES
   Monterey, CA 93943
   darken@nps.navy.mil

5. Prof. Mike Zyda
   Code MOVES
   Monterey, CA 93943
   zyda@nps.navy.mil